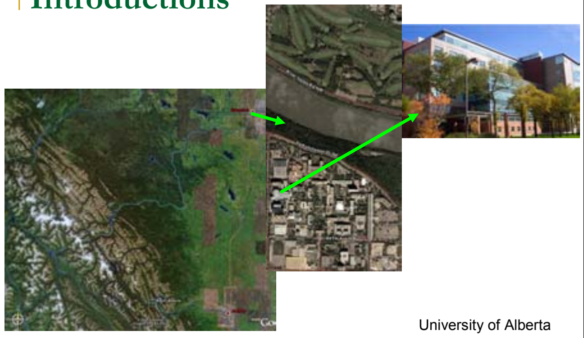
Discovering Structure in Data

Fast Classification Using the DataSqueezer Algorithm

Lukasz Kurgan
University of Alberta

Introductions



University of Alberta
Department of Electrical and Computer Engineering
data mining with applications to computational biology

Discovering Structure in Data

- **Tabular (relational) multi-attribute and multi-sample**
  - **e.g. clinical patient records, micro-array data, protein sequence data banks…**
  - **Numerical and nominal values**
- **Highly dimensional**
  - **# data samples (few thousands to few millions, or more…)**
  - **# attributes (few to several hundred, or more…)**
- **Analysis of such data is possible only using automated computational methods**

Discovering Structure in Data

- **Data Mining**
  - **defined as extraction of valid, useful, easily understandable knowledge from large collections of data, for high level decision making**
  - **research interests**
    - **data preprocessing (discretization, missing data imputation)**
    - **automated generation of data models**
      - **production and association rules**
    - **classification**
      - **discrete target concept**
    - **prediction**
      - **continuous target concept**

Discovering Structure in Data

- automated generation of data models
  - does not require restrictive statistical assumptions such as independence, linear relationships, multi-colinearity, normality, etc.
  - finds rules for which a set of (independent) variables are correlated with a result, which simply means that given the 'IF' condition, the 'THEN' result occurs a given percentage of time.

target

DECISION	CONDITION1	CONDITION2	INDEX
A	low	normal	2
A	low	normal	3
A	normal	normal	3
A	normal	low	2
A	normal	low	1
B	low	high	4
B	low	low	4
B	high	normal	4
A	normal	low	2
A	normal	normal	2
A	normal	normal	2
A	normal	normal	3
A	low	normal	1
B	normal	high	4
B	high	low	4
B	high	normal	4
A	normal	low	4
A	low	normal	2
A	low	normal	2
A	normal	normal	4
A	normal	normal	2
B	low	high	4
B	high	low	4
B	high	normal	4

Discovering Structure in Data

- automated generation of data models
  - does not require restrictive statistical assumptions such as independence, linear relationships, multi-colinearity, normality, etc.
  - finds rules for which a set of (independent) variables are correlated with a result, which simply means that given the 'IF' condition, the 'THEN' result occurs a given percentage of time.

~~target~~

DECISION	CONDITION1	CONDITION2	INDEX
A	low	normal	2
A	low	normal	3
A	normal	normal	3
A	normal	low	2
A	normal	low	1
B	low	high	4
B	low	low	4
B	high	normal	4
A	normal	low	2
A	normal	normal	2
A	normal	normal	2
A	normal	normal	3
A	low	normal	1
B	normal	high	4
B	high	low	4
B	high	normal	4
A	normal	low	4
A	low	normal	2
A	low	normal	2
A	normal	normal	4
A	normal	normal	2
B	low	high	4
B	high	low	4
B	high	normal	4

RULES for DECISION: A (4 rules)
1. **IF CONDITION1 = normal AND CONDITION2 = normal THEN DECISION = A**
2. IF INDEX = 2 THEN DECISION = A
3. IF CONDITION1 = normal AND CONDITION2 = low THEN DECISION = A
4. IF CONDITION1 = low AND CONDITION2 = normal THEN DECISION = A

RULES for DECISION: B (3 rules)
1. **IF CONDITION1 = high AND INDEX = 4 THEN DECISION = B**
2. IF CONDITION1 = low AND INDEX = 4 THEN DECISION = B
3. IF CONDITION1 = normal AND CONDITION2 = high AND INDEX = 4 THEN DECISION = B

Discovering Structure in Data

- automated generation of data models
  - does not require restrictive statistical assumptions such as independence, linear relationships, multi-colinearity, normality, etc.
  - finds rules for which a set of (independent) variables are correlated with a result, which simply means that given the 'IF' condition, the 'THEN' result occurs a given percentage of time.

DECISION	CONDITION1	CONDITION2	INDEX
A	low	normal	2
A	low	normal	3
A	normal	normal	3
A	normal	low	2
A	normal	low	1
B	low	high	4
B	low	low	4
B	high	normal	4
A	normal	low	2
A	normal	normal	2
A	normal	normal	2
A	normal	normal	3
A	low	normal	1
B	normal	high	4
B	high	low	4
B	high	normal	4
A	normal	low	4
A	low	normal	2
A	low	normal	2
A	normal	normal	4
A	normal	normal	2
B	low	high	4
B	high	low	4
B	high	normal	4

ASSOCIATIONS (no "decision" target)
1. DECISION = B AND INDEX = 4
2. CONDITION1 = normal AND CONDITION2 = normal AND INDEX = 2
3. CONDITION2 = normal AND INDEX = 3
4. DECISION = A AND INDEX = 2
etc.

Discovering Structure in Data

- **data models (rules, and others)**
  - **can be generated very fast**
    - **log-linear time with respect to number of data points**
  - **associations and rules allow to find hidden relations**
  - **rules (and other models) allow to perform classification and prediction**
    - **both associations (a special type called association classification) and rules can be used**
    - **other models include: decision trees, bayesian, regression, neural networks, support vector machines, instance-based,… (may require more computations)**

Discovering Structure in Data

- **relevance**
  - **biology is a source of large and often unexplored databases**
  - **many biological problems can be translated into model generation and analysis, prediction and/or classification tasks**
    - **the goal is to find structure in the data**
  - **my recent research interests are in**
    - **analysis of clinical data to discover disease biomarkers, new treatments and diagnostic procedures**
    - **protein structure analysis and prediction; analysis of both individual proteins and large protein clusters based on data stored in protein data banks**

DataSqueezer Algorithm

- comparison of fast rule learners

learner	Complexity	reference	learner	Complexity	reference
REP	$O(s^4)$	Cohen, 1993	**RIPPER**	nearly linear complexity, not worse than O(s logs)	Cohen, 1995
C4.5 rules	$O(s^3)$	Cohen, 1995			
LERILS	$O(s^2)$	Chisholm and Tadepalli, 2002	**SLIPPER**	nearly linear complexity, not worse than O(s logs)	Cohen and Singer, 1999
RISE	$O(s^2)$	Domingos, 1994			
CN2	$O(s^2)$	Domingos, 1994	IREP++	nearly linear complexity, not worse than O(s logs)	Dain, Cunningham and Boyer, 2004
CLIP4	$O(s^2)$	Cios and Kurgan, 2004			
DiVS	$O(s^2)$	Sebag, 1996	**C5.0**	nearly linear complexity, not worse than O(s logs)	Cohen and Singer, 1999
IREP	$O(s \log^2 s)$	Furnkranz and Widmer, 1994			

DataSqueezer Algorithm

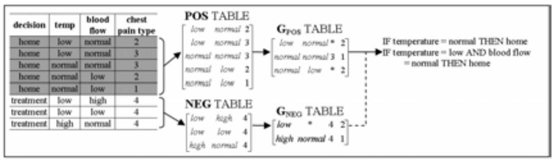
```
Given:    POS, NEG, k (number of attributes), s (number of examples)
Step1.
1.1       G_POS = DataReduction(POS, k);
1.2       G_NEG = DataReduction(NEG k);

Step2.
2.1       Initialize RULES = []; i=1;                      // where rules_i denotes i^th rule stored in RULES
2.2       create LIST = list of all columns in G_POS
2.3       within every G_POS column that is on LIST, for every non missing value a from selected column j compute sum, s_aj, of values of
          gpos_i[k+1] for every row i, in which a appears and multiply s_aj, by the number of values the attribute j has
2.4       select maximal s_aj, remove j from LIST, add "j = a" selector to rules_i
2.5.1     if rules_i does not describe any rows in G_NEG
2.5.2         then remove all rows described by rules_i from G_POS, i=i+1;
2.5.3           if G_POS is not empty go to 2.2, else terminate
2.5.4       else go to 2.3
Output:   RULES describing POS

          DataReduction (D, k)                             // data reduction procedure for D=POS or D=NEG
DR.1      Initialize G = []; i=1; tmp = d_1; g_1 = d_1; g_1[k+1]=1;
DR.2.1    for j=1 to N_D                                   // for positive/negative data; N_D is N_POS or N_NEG
DR.2.2      for kk = 1 to k                                // for all attributes
DR.2.3        if (d_j[kk] ≠ tmp[kk] or d_j[kk] = '*')
DR.2.4          then tmp[kk] = '*';                        // '*' denotes missing" do not care" value
DR.2.5      if (number of non missing values in tmp ≥ 2)
DR.2.6          then g_i = tmp; g_i[k+1]++;
DR.2.7        else i++; g_i = d_j; g_i[k+1]=1; tmp =  d_j;
DR.2.8    return G;
```

DataSqueezer Algorithm

- example



decision	temp	blood flow	chest pain type
home	low	normal	2
home	low	normal	3
home	normal	normal	3
home	normal	low	2
home	normal	low	1
treatment	low	high	4
treatment	low	low	4
treatment	high	normal	4

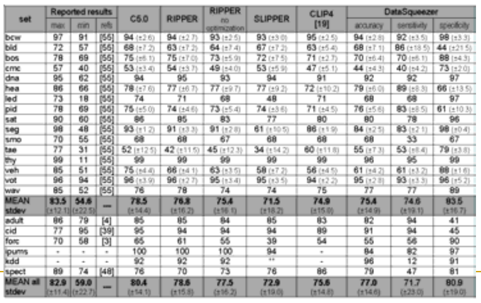
IF temperature = normal THEN home
IF temperature = low AND blood flow = normal THEN home

DataSqueezer Algorithm

- **DataSqueezer was extensively tested to shows:**
  - **competitiveness with other state-of-the-art rule learners in the accuracy and complexity of the rules it generates**
  - **better scalability**
    - **the empirical complexity of DataSqueezer closely matches the calculated log-linear asymptotic complexity, while the running time for DataSqueezer is far shorter than for other competing learners**
  - **good robustness to missing data**

DataSqueezer Algorithm

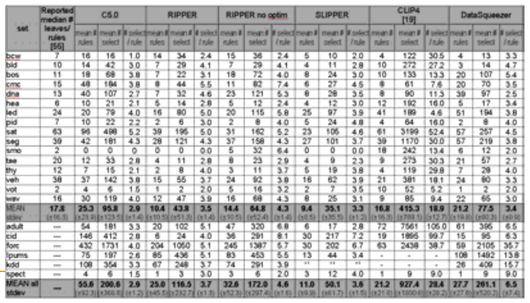
- accuracy



set	Reported results			C5.0	RIPPER	RIPPER no optimization	SLIPPER	CLIP4 [19]	DataSqueezer		
	max	min	refs						accuracy	sensitivity	specificity
bcw	97	91	[55]	94 (±2.6)	94 (±2.7)	93 (±2.5)	93 (±3.0)	95 (±2.5)	94 (±2.8)	92 (±3.5)	98 (±3.3)
bld	72	57	[55]	68 (±7.2)	63 (±7.2)	64 (±7.4)	67 (±7.2)	63 (±5.4)	68 (±7.1)	86 (±18.5)	44 (±21.5)
bos	78	69	[55]	75 (±6.1)	75 (±7.0)	73 (±5.9)	72 (±7.5)	71 (±2.7)	70 (±6.4)	70 (±6.1)	88 (±4.3)
cmc	57	40	[55]	53 (±3.4)	54 (±3.7)	49 (±4.0)	53 (±5.9)	47 (±5.1)	44 (±4.3)	40 (±4.2)	73 (±2.0)
dna	95	62	[55]	94	95	93	94	91	92	92	97
hea	86	66	[55]	78 (±7.6)	77 (±6.7)	77 (±9.7)	77 (±9.2)	72 (±10.2)	79 (±6.0)	89 (±8.3)	66 (±13.5)
led	73	18	[55]	74	71	68	48	71	68	68	97
pid	78	69	[55]	75 (±5.0)	74 (±4.6)	73 (±5.4)	74 (±3.6)	71 (±4.5)	76 (±5.6)	83 (±8.5)	61 (±10.3)
sat	90	60	[55]	86	85	83	77	80	80	78	96
seg	98	48	[55]	93 (±1.2)	91 (±3.3)	91 (±2.8)	61 (±10.5)	86 (±1.9)	84 (±2.5)	83 (±2.1)	98 (±0.4)
smo	70	55	[55]	68	68	67	68	68	68	33	67
tae	77	31	[55]	52 (±12.5)	42 (±11.5)	45 (±12.3)	34 (±14.2)	60 (±11.8)	55 (±7.3)	53 (±8.4)	79 (±3.8)
thy	99	11	[55]	99	99	99	99	99	96	95	99
veh	85	51	[55]	75 (±4.4)	66 (±4.1)	63 (±3.5)	58 (±7.2)	56 (±4.5)	61 (±4.2)	61 (±3.2)	88 (±1.6)
vot	96	94	[55]	96 (±3.9)	96 (±2.7)	95 (±3.4)	95 (±3.5)	94 (±2.2)	95 (±2.8)	93 (±3.3)	96 (±5.2)
wav	85	52	[55]	76	78	74	74	75	77	77	89
MEAN stdev	**83.5** (±12.1)	**54.6** (±22.5)	—	**78.5** (±14.4)	**76.8** (±16.2)	**75.4** (±16.1)	**71.5** (±18.2)	**74.9** (±15.0)	**75.4** (±14.9)	**74.6** (±19.1)	**83.5** (±16.7)
adult	86	79	[4]	85	85	84	85	83	82	94	41
cid	77	95	[39]	95	94	94	94	89	91	94	45
forc	70	58	[3]	65	61	55	39	54	55	56	90
ipums	-	-	-	100	100	100	94	-	84	82	97
kdd	-	-	-	92	92	92	**	-	96	12	91
spect	89	74	[48]	76	70	73	76	86	79	47	81
MEAN all stdev	**82.9** (±11.4)	**59.0** (±22.7)	—	**80.4** (±14.1)	**78.6** (±15.8)	**77.5** (±16.2)	**72.9** (±19.0)	**75.6** (±14.8)	**77.0** (±14.6)	**71.7** (±23.0)	**80.9** (±19.0)

DataSqueezer Algorithm

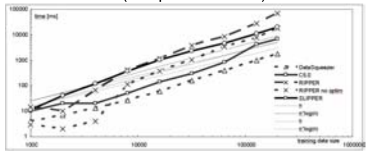
- rules



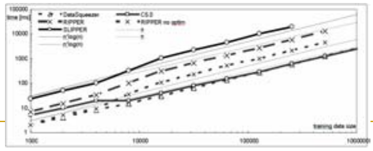
set	Reported median # leaves/ rules [55]	C5.0			RIPPER			RIPPER no optim			SLIPPER			CLIP4 [19]			DataSqueezer		
		mean # rules	mean # select	# select / rule	mean # rules	mean # select	# select / rule	mean # rules	mean # select	# select / rule	mean # rules	mean # select	# select / rule	mean # rules	mean # select	# select / rule	mean # rules	mean # select	# select / rule
bcw	7	16	16	1.0	14	34	2.4	15	36	2.4	5	10	2.0	4	122	30.5	4	13	3.3
bld	10	14	42	3.0	7	29	4.1	7	29	4.1	4	11	2.8	10	272	27.2	3	14	4.7
bos	11	18	68	3.8	7	22	3.1	18	72	4.0	8	24	3.0	10	133	13.3	20	107	5.4
cmc	15	48	184	3.8	8	44	5.5	11	82	7.4	6	27	4.5	8	61	7.6	20	70	3.5
dna	13	40	107	2.7	7	32	4.6	23	121	5.3	8	28	3.5	8	90	11.3	39	97	2.5
hea	6	10	21	2.1	5	14	2.8	5	12	2.4	4	12	3.0	12	192	16.0	5	17	3.4
led	24	20	79	4.0	16	80	5.0	20	115	5.8	25	97	3.9	41	189	4.6	51	194	3.8
pid	7	10	22	2.2	2	6	3.0	2	8	4.0	5	24	4.8	4	64	16.0	2	8	4.0
sat	63	96	498	5.2	39	196	5.0	31	162	5.2	23	106	4.6	61	3199	52.4	57	257	4.5
seg	39	42	181	4.3	28	121	4.3	37	158	4.3	27	101	3.7	39	1170	30.0	57	219	3.8
smo	2	0	0	0	0	0	0.0	5	32	6.4	0	0	0.0	18	242	13.4	6	12	2.0
tae	20	12	33	2.8	4	11	2.8	8	23	2.9	4	9	2.3	9	273	30.3	21	57	2.7
thy	12	7	15	2.1	2	8	4.0	3	11	3.7	5	19	3.8	4	119	29.8	7	28	4.0
veh	38	37	142	3.8	15	55	3.7	24	92	3.8	16	62	3.9	21	381	18.1	24	80	3.3
vot	2	4	6	1.5	1	2	2.0	5	16	3.2	2	7	3.5	10	52	5.2	1	2	2.0
wav	16	30	119	4.0	12	47	3.9	16	68	4.3	8	25	3.1	9	85	9.4	22	65	3.0
MEAN stdev	**17.8** (±16.3)	**25.3** (±23.9)	**95.8** (±123.5)	**2.9** (±1.4)	**10.4** (±10.5)	**43.8** (±51.3)	**3.5** (±1.4)	**14.4** (±10.5)	**64.8** (±52.4)	**4.3** (±1.4)	**9.4** (±8.5)	**35.1** (±35.5)	**3.3** (±1.2)	**16.8** (±16.3)	**415.3** (±789.1)	**18.9** (±12.7)	**21.2** (±19.8)	**77.5** (±80.3)	**3.4** (±0.9)
adult	---	54	181	3.3	20	102	5.1	47	320	6.8	6	17	2.8	72	7561	105.0	61	395	6.5
cid	---	146	412	2.8	6	24	4.0	36	291	8.1	30	217	7.2	19	1895	99.7	15	95	6.3
forc	---	432	1731	4.0	204	1050	5.1	245	1387	5.7	30	202	6.7	63	2438	38.7	59	2105	35.7
ipums	---	75	197	2.6	85	436	5.1	83	453	5.5	13	44	3.4	-	-	-	108	1492	13.8
kdd	---	108	354	3.3	67	248	3.7	74	291	3.9	**	**	**	-	-	-	26	409	15.7
spect	---	4	6	1.5	1	3	3.0	3	6	2.0	3	12	4.0	1	9	9.0	1	9	9.0
MEAN all stdev	---	**55.6** (±92.3)	**200.6** (±368.6)	**2.9** (±1.2)	**25.0** (±45.5)	**116.5** (±232.7)	**3.7** (±1.3)	**32.6** (±52.3)	**172.0** (±297.4)	**4.6** (±1.6)	**11.0** (±9.9)	**50.1** (±61.7)	**3.6** (±1.5)	**21.2** (±21.8)	**927.4** (±1800.6)	**28.4** (±28.2)	**27.7** (±27.6)	**261.1** (±520.2)	**6.5** (±7.4)

DataSqueezer Algorithm

- computational complexity
  - real-life dataset (complex rule base)



  - synthetic dataset (compact rule base)



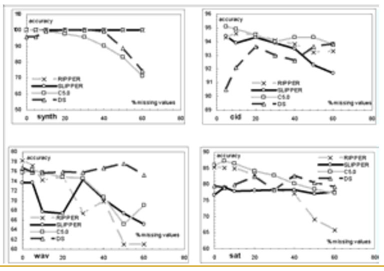
DataSqueezer Algorithm

- **computational complexity**
  - **based on the linear approximations of the results, an extrapolation of the running time for a larger (20 million examples) real-life dataset was computed. It would require about**
    - **27 minutes for DataSqueezer**
    - **1 hour 45 minutes for C5.0**
    - **4 hours and 18 minutes for RIPPER without optimization**
    - **5 hours 15 minutes for RIPPER**
    - **16 hours 20 minutes for SLIPPER**

    to perform induction using the same hardware.

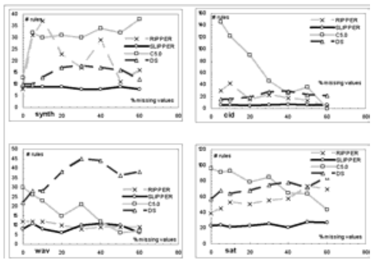
DataSqueezer Algorithm

- robustness to missing data (accuracy)



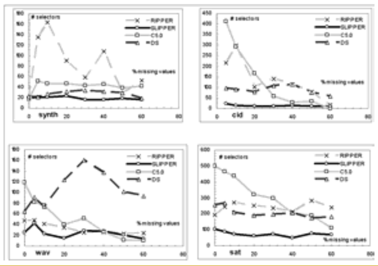
DataSqueezer Algorithm

- robustness to missing data (number of rules)



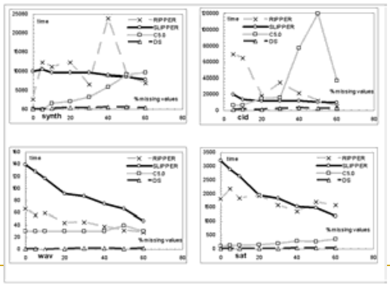
DataSqueezer Algorithm

- robustness to missing data (number of selectors)



DataSqueezer Algorithm

- robustness to missing data (running time)



DataSqueezer Algorithm

- **robustness to missing data**
  - **DataSqueezer shows superior robustness to missing values in terms of**
    - **running time**
    - **stable level of accuracy**

    and fairly good robustness in terms of number and complexity of generated rules.
  - **The other robust learner is SLIPPER, which shows superior properties in terms of**
    - **the number and complexity of generated rules**
    - **high and fairly stable accuracy of generated rules**

DataSqueezer Algorithm

- **Summary**
  - **simple to implement**
  - **generates rules with accuracy and complexity comparable to the rules generated by other learners**
  - **characterized by log-linear complexity combined with speed that is superior when compared with other scalable, modern learners**
  - **superior robustness to missing values**

Kurgan L, Cios K and Dick S, Highly Scalable and Robust Rule Learner: Performance Evaluation and Comparison, *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 36, no. 1, pp. 32-53, 2006

Example Application

- **Analysis of Cystic Fibrosis (CF) data**
  - **CF is a deadly genetic disease; it affects respiratory system, digestive system, endocrine system, and reproductive system**
  - **Project involved analysis of clinical CF data**
    - **in collaboration between the University of Colorado and the Denver's Children Hospital**
    - **(temporal) data on 856 patients collected starting in 1982**
  - **Goals**
    - **discovery of important factors that influence the pace of development of CF**
      - **several categories were defined based on an attribute that quantifies the progress of the disease in terms of the respiratory functions**
    - **discovery of important factors that are related to particular kinds of CF**
      - **CF is caused by at least 500 different genetic mutations but approximately 70% of the mutations are found to be "delta F508" gene (the most common CF mutation)**
      - **three kinds of CF were defined and analyzed: 1) both, Genotype 1 and Genotype 2 are F508, 2) either Genotypes 1 or Genotype 2 is F508, and the other is any other genotype, 3) both Genotype 1 and Genotype 2 are not F508**

Example Application

- Analysis of CF data
  - sample results for goal 1
    - significant and previously unknown finding was a relation between high value of sweatelectr1 (potassium levels) and the improvement of the disease

ATRIBUTE	VALUE	MARK	FASTDEGRAD					IMPROV					NOCHANGE					SLOWDEGRAD				
			TI1	TI2	TI3	TI4	TI5	TI1	TI2	TI3	TI4	TI5	TI1	TI2	TI3	TI4	TI5	TI1	TI2	TI3	TI4	TI5
CFtypes (cf)	Type4	2+										▨										▨
race (dem)	Black	3+						▨		▨	▨											
group (dem)	C	3+/4+	▨	▨		▨										▨		▨	▨			
group (dem)	NBS	3+/4+	▨					▨	▨	▨	▨	▨	▨	▨	▨	▨		▨			■	
group (dem)	MI	3+/4+						▨	▨													
group (dem)	FN	3+/4+							▨	▨												
motage (dem)	[22.50,48.50)	3+	▨			▨		▨			▨	▨	▨	▨	▨	▨	▨	▨			▨	
motage (dem)	[19.50,22.50)	3+			▨														▨			
mecil (dem)	TreatedSurgically	2+				▨																
sweatelectr1 (dia)	**[24.50,46.00)**	**4+**								▨	▨	▨	▨	▨		▨						
sweatelectr2 (dia)	[-11.00,95.50)	3+						▨	▨		▨											
tobraresistent? (cul)	Suscept.	2+																		▨		
na (mic)	[129.00,143.00)	2+																			▨	
prot (mic)	[-1.95,7.95)	2+																			▨	
vita (mic)	[0.44,748.00)	2+																			▨	
wbc (hem)	[4.05,18.00)	2+																			▨	
hct (hem)	[27.40,45.50)	2+																			▨	
mch (hem)	[24.90,91.40)	2+																			▨	
mchc (hem)	[30.40,35.80)	2+																			▨	
rdw (hem)	[-0.85,15.40)	2+																			▨	
HAZ (per)	[-2.91,-1.87)	3+	▨																			
age@diagnosis (dia)	025till2	3+		▨	▨		▨												▨	▨		

Kurgan L., Cios K., Sontag M., and Accurso F., Mining the Cystic Fibrosis Data, In: Zurada J. and Kantardzic M., (Eds.), *Next Generation of Data-Mining Applications*, pp. 415-444, IEEE Press - Wiley (ISBN 0-471-65605-4), 2005

THANK YOU